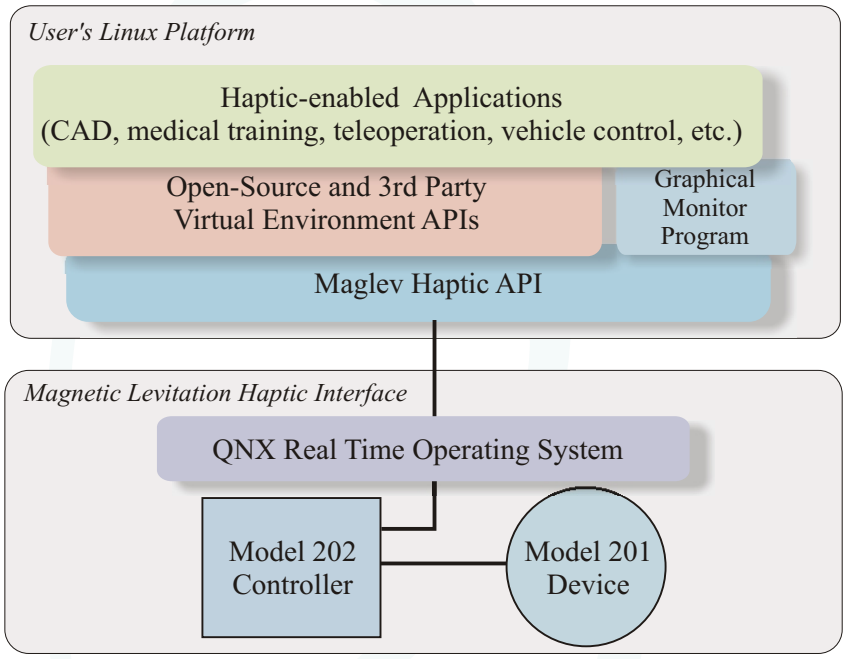
Butterfly Haptics

Maglev 200™ Magnetic Levitation Haptic Interface

System relationships

In the Maglev 200™ system, critical real-time processes take place in the Model 202 Controller running the rock-solid QNX™ 6.3 hard RTOS, freeing the user's platform to concentrate on the physics and graphics part of the application. The result is enhanced performance and greater ease of use. Connection between the user's platform (client) and maglev system (server) is by Ethernet. Analog actuator and sensor cables connect between the device and controller.



The user may program the maglev system by linking to the provided C/C++ Maglev Haptic API or by using other open-source or 3rd party virtual environment APIs as well as systems such as MATLAB® .

The Maglev Haptic API supports multiple concurrent clients and multiple maglev haptic systems. A unique included Graphical Monitor Program gives direct access to most of the API functions, allowing straightforward system monitoring and application debugging.

Application Programming Interface

We recognize that users want full access to all hardware functions of the maglev system. The Maglev Haptic API provides this very low level of functionality in addition to a large number of higher-level convenience functions running on the "server side," relieving the user's application from needing to perform many burdensome tasks. Numerical quantities in the API are in SI units, making it straightforward to relate to physical quantities in the user's virtual or remote environment. The API function groups are shown below.

Haptic interaction functions

* Reading positions and orientations
* Applying forces and torques
* Reading translational and rotational velocities
* Reading button status
* Setting the haptic loop frequency

Callback functions

* Registering a tick timing callback
* Registering a boundary violation callback
* Registering a button pressed callback
* Registering a fault condition callback

High level convenience functions

* Automatic takeoff and landing
* Automatic gravity finding and cancellation
* Setting a user coordinate frame
* Locking or constraining individual axes
* Setting a spherical workspace boundary
* Setting gains for normal, constrained axes, locked axes, and spherical boundary

Robotic functions

* Setting translational and rotational velocities
* Setting position and orientation

Butterfly Haptics

SOFTWARE

Maglev 200™ Magnetic Levitation Haptic Interface

Application Programming Interface (cont.)

Low level functions

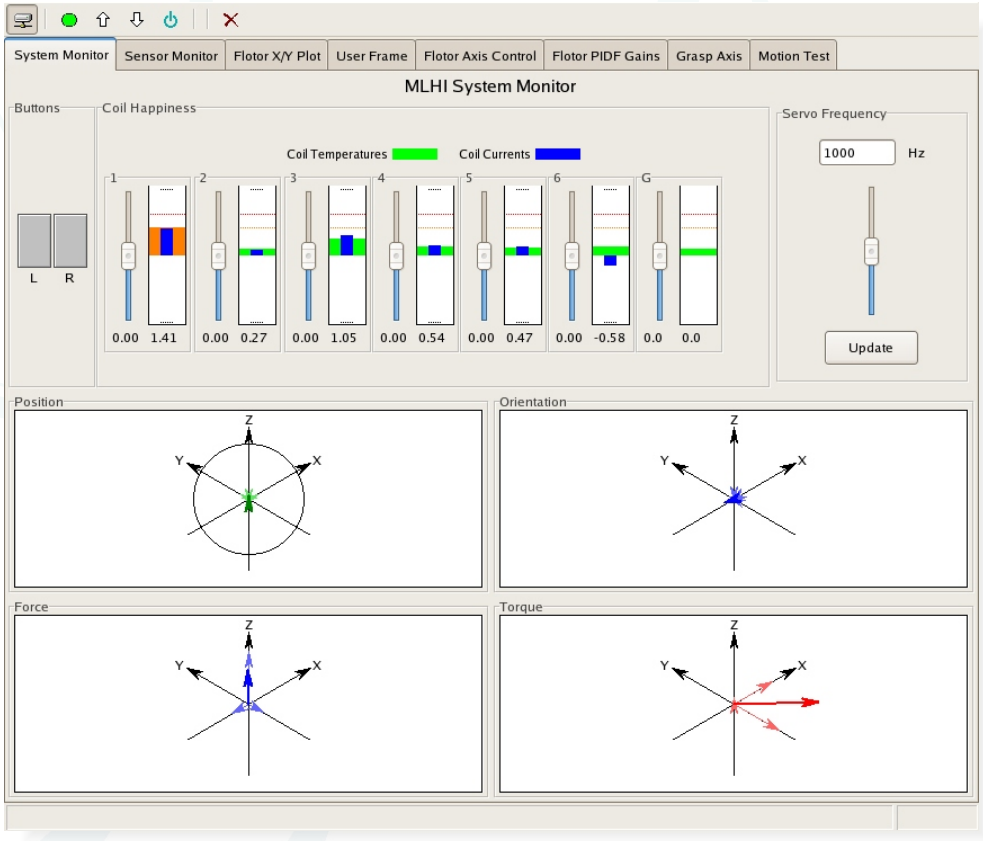
* Reading sensor voltages
* Reading sensor position values
* Reading and setting coil currents
* Setting coil current limits
* Setting velocity limits

Miscellaneous functions

* Reading computed coil temperatures
* Connecting and disconnecting to/from host
* System shutdown

Graphical Monitor Program

The included Graphical Monitor Program, source code and binary, provides a quick and easy way to exercise virtually all of the Maglev Haptic API, giving the user immediate access to the hardware functionality without writing a line of code.



For example, the lower part of the MLHI System Monitor panel shown here graphically displays the position, orientation, force, and torque exerted on the handle. Button status, coil currents, and coil temperatures are displayed in the upper part. Sliders permit currents to be set in individual coils. Finally, the servo frequency can be set.

Other panels, accessed from the tabs at the top, allow the user to see low-level sensor data, plot a position trajectory in the XY plane, set up a user-defined frame, control individual axes by locking their positions or constraining their motions between limits, set proportional, integral, derivitive, and feedforward controller gains, control an optional grasp axis, or generate sine wave motion in all axes.

The Graphical Monitor Program makes it easy to try out operation with reduced degrees of freedom by locking unwanted axes. Virtual object stiffnesses and stability margins can be quickly evaluted to find correct values for incorporation in the user's code. Since the Graphical Monitor Program can run concurrently with the user's application, it becomes a straightforward matter to diagnose programming or hardware issues. The net result is to make a complex and cutting edge technology much easier to use.